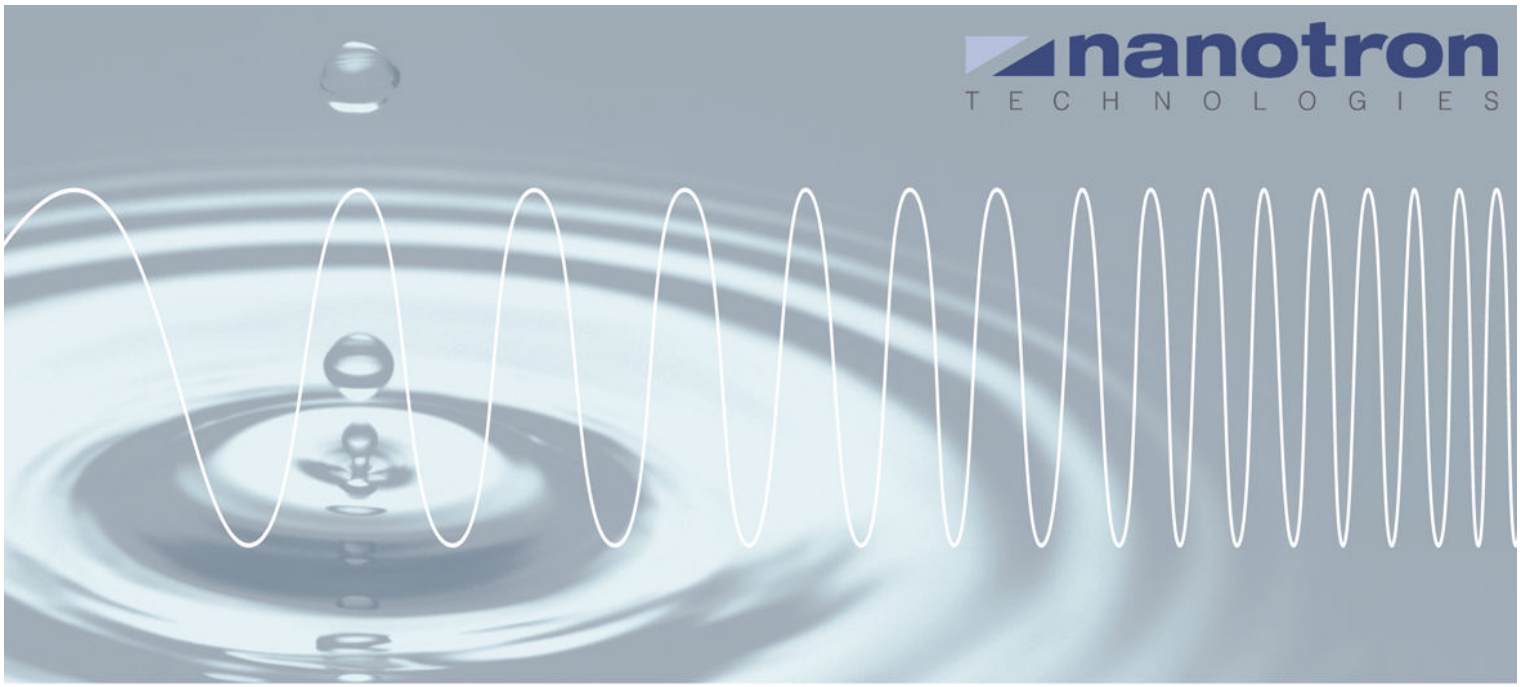

swarm API

1.6.1

NA-13-0267-0003-1.6.1

Document Information

Document Title:	*swarm API*
Document Version:	1.6.1
Current Date:	2013-02-21
Print Date:	2013-02-21
Document ID:	NA-13-0267-0003-1.6.1
Document Author:	Dr. Frank Schlichting

Contents

1. Scope

Scope of this document is to define a hardware independent Application Programming Interface (API) to realize the low level ranging functionality for a *swarm* radio described in chapter 2. A swarm is defined as a congregation of independent radios or nodes which share a common interest in their relative positioning and communication towards each other for a certain period of time.

Main focus on creating the API is to minimize effort in order to

a) implement ultimate mobility of any *swarm* of nodes
b) support development of *swarm* applications
c) fuel *swarm* productization a migration to volume solutions.

with best possible basic ranging performance. To easily accomplish these tasks this is supported by

a) using existing Nanotron hardware components and interfaces,
b) providing a generally available HW interface that can easily be controlled by a host platform,
c) breaking down the functionality into small functional elements on the embedded part, and
d) realizing more complex system functionality on the customers' host platform allowing for maximum flexibility for the intended appication.

2. Application

The intended application is a multi node peer to peer ranging solution in which every embedded *swarm* radio is connected to a host platform. Transceiver nodes capable of ranging and communicating with other nodes are controlled through their API by a host platform. The individual host platform controls the embedded platform to range to one of the remaining other nodes and returns the distance value between the two nodes.

Three different types of *swarm* behaviour exist: ACTIVE, PASSIVE and SNIFFER

ACTIVE: A node with active behaviour is able to initiate a ranging as well as a communication operation to other nodes with both active and passive behaviour. It responds to ranging requests of other nodes and forwards results to the controlling host platform. All parameters of an active platform can be controlled via the API. A node with active behaviour can also act as a passive node.

PASSIVE: A node with passive behaviour responds to ranging requests of other nodes. It cannot initiate a ranging or a communication operation to other *swarm* radios. Since there is no controlling host platform it does not forward ranging results. All parameters of a passive platform are fixed and cannot be controlled via an API. However, an active node behaving as a passive node can be switched back to its active behaviour.

SNIFFER: A node with sniffer behaviour listens to all radio communication between active and passiv as well as between active and active nodes. It forwards the received packets to its host platform via the API. A sniffer node cannot initiate a ranging or communication operation to other *swarm* nodes. It does not respond to ranging requests of other nodes.

All nodes with active behaviour are equal and all nodes are independent. Therefore all active nodes shall obtain the full ranging information. The higher level application layer is not part of the API specification and will be programmed by the customer. The host platform will be able to select the NODE ID of the partner node, which corresponds to the tag's MAC address, it wants to range to.

A veriety of hardware platforms will be available supporting specific requirements, for instance higher precision ranging, dedicated development support, specific form factors, etc. All these *swarm* platforms will support the general functionality of this API supporting hardware independent application development.

The *swarm* high precision radios will feature two switchable antennas. The host platform is able to select which antenna shall be used on the embedded platform to perform the ranging. Alternatively antenna diversity can be switched on by the host. Then the embedded node will autonomously perform two consecutive ranging operations to the selected partner node – one for each node antenna. In this case the smaller of the two obtained ranging values will be returned to the host. This ensures significant improvement on possible multi path effects.

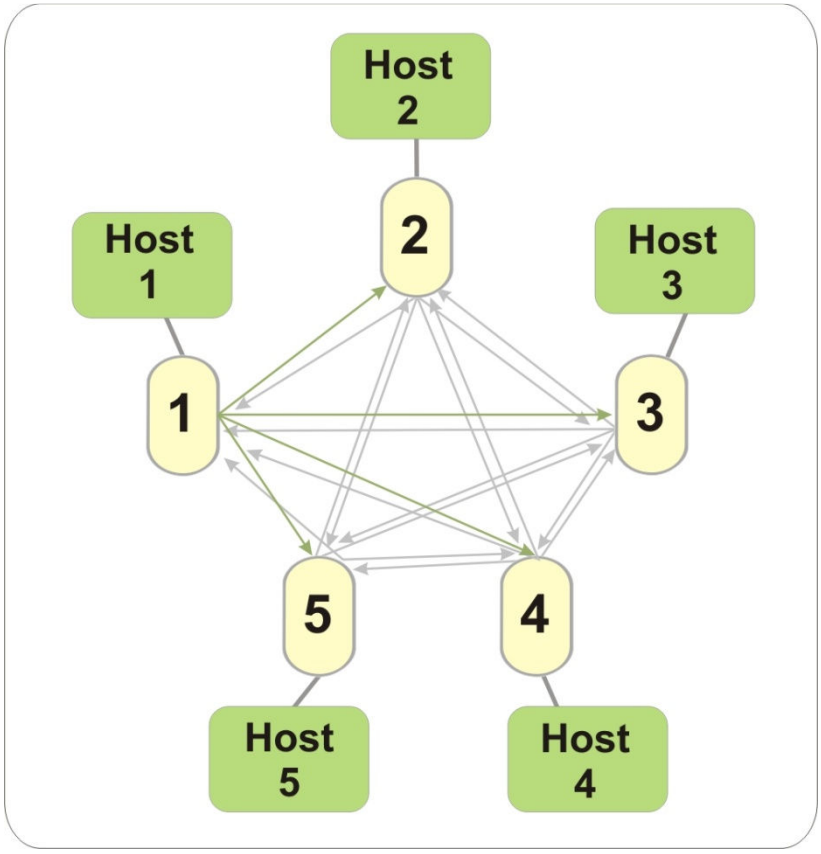

Fig.1: Principle of a *swarm* kit – here: for 5 *swarm* radios

All nodes shall be independent from each other and also be battery operated to allow fully mobile applications. It shall be possible to identify new nodes by listening to the node's ID broadcast paket or other ranging operations already performed by this node. For this purpose the *swarm* radio shall be able to perform two tasks:

1. Send out ID broadcast information. This feature can be deactivated.
2. Listening to other broadcast IDs and storing the respective IDs in a *swarm* ID list which can be read out by the host.

Furthermore it shall be possible to communicate a data packet of variable length peer to peer from one active *swarm* radio A to another active *swarm* radio B.

Swarm does support active and passive behaviour (see above). Active nodes can be accessed via the API to control data and ranging operations. They do report ranging results as well as received *swarm* node IDs in an ID list.
Passive nodes respond to a ranging request and send out their blink ID. However, passive notes are not able to be controlled via the API and will not report ranging results and communication data.

Data communication can only be performed between active nodes. The node's behaviour – active or passive – is coded in the node's blink ID packet which is broadcasted to all other nodes.

The automatic ranging response for active packets can be deactived ("privacy mode").

3. Hardware Platforms & Interface to Host

A variety of different *Swarm* radios exist. The description of both the embedded *swarm* radio hardware and its interface to the host controller is described in a separate document for each type of *swarm* radio. Please refer to the respective manual to learn about the details for the individual radio. The API described in this document is valid for all types of *swarm* radios. However, not all radios support the full functionality of this API, e.g. antenna diversity is only supported by *Swarm* radios high precision.

4. Application Programming Interface

4.1. General Requirements

The following general software requirements have to be met for implementing the embedded software code for *swarm* functionality:

- Peer to peer ranging: 1 to N, Addressing: Node ID
- Low level API based on single ranging
- API returns ranging value or error code after timeout
- UART interface to host for each unit
- Maximum time for one elementary ranging operation: <=12.5 ms
- No power down required
- Media access mode ALOHA, optionally CSMA
- Diversity on/off configuration
- Diversity switching on initiating unit only

4.2. General Communication Protocol & Host vs. *swarm* Synchronisation

For the general communication protocol the following conventions apply:

1. All communication via the interface is done by ASCII characters. This implies that e.g. a 6 byte node ID (hexadecimal) will be transmitted in the following format:

Node ID (hex)												
0000BF260468	0	0	0	0	B	F	2	6	0	4	6	8
ASCII (hex)	30	30	30	30	42	46	32	36	30	34	36	38

2. All command communication ends with carriage return / line feed:

Command termination			
… \r\n	…	\r	\n
ASCII (hex)	…	0D	0A

3. All command codes and their respective parameters are separated by one space character (ASCII 20)

   Example: `RangeTo 0000BF260468`

4. All commands are transmitted MSB first, LSB last.

5. Return code for unknown or erroneous command is „ERR\r\n“

   Example:
   ```
   WrongCommand xyz
   ERR<CR, LF>
   ```

4.3. API Command Set Overview

This chapter summarizes and catagorizes the API Command Set which is available to interact with the embedded ranging hardware platform:

4.3.1. *swarm* radio Setup Commands

SetNodeIDAdd	Sets the Node ID of *swarm* node
ReadNodeIDAdd	Readback of configured Node ID of node connected to host
SetNodeType	Sets the type of node which will also be broadcasted in ID blinks
SaveSettings	Saves all setting including Node ID permanently to EEPROM
RestoreSettings	Restores all parameter settings from EEPROM
ReadSettings	Readback of current Node configuration
SetFactorySettings	Reset device configuration to factory default settings

4.3.2. Ranging Commands

SetPrivacyMode	Enables and disables response to a received ranging request
RangeTo	Initiates an elementary ranging cycle to another *swarm* node
GetRangingResults	Reports the received indirect ranging results between the selected node <NodeID> and other nodes with IDs <TargetID 1>, <TargetID 2>, … <TargetID n> which have a maximum age of AGE in seconds
BroadcastRangingResults	Enables/Disables the broadcast transmission of ranging results after each successful ranging.
DeleteAllRangingResults	Deletes all entries in Ranging Results List

4.3.3. Data Communication Commands

EnableDataNotification	Enables and disables data notification
SendDataTo	Sends <data> of length <len> to node <ID>
GetData	Reads out transmitted data
BroadcastData	Broadcasts <data> of length <len> to all nodes
EnableRangingData	Enables and disables the transmission of data from the ranging data buffer along with a ranging operation initiated by RangeTo.
FillRangingData	Fills the ranging data buffer with <data> of length <len>. This data will be transmitted with the next RangeTo operation if EnableRangingData is <On>. The ranging data <data> is contained within the ranging packet itself.

4.3.4. *swarm* radio Node Identification

SetBroadcastNodeID	Enables and disables broadcast of Node ID blink packets
SetBroadcastInterval	Sets the broadcast interval in which the Node ID will be sent
GetNodeIDList	Reports the currently valid NodeIDList with time stamps

4.3.5. Air Interface Commands

SetCSMA	Switches CSMA mode on and off and determines back-off factor for CSMA
SetDiversity	Switches diversity mode for this node on and off
SetAntenna	Selects the active antenna to be used for ranging operation if diversity is off

4.4. API Command Set

In order to interact with the embedded ranging hardware platform the following API command set is implemented:

4.4.1. *swarm* radio Setup Commands

SetNodeIDAdd <ID>:

Description: Sets the Node ID of *swarm* node to <ID>
Parameters: <ID>
Format: 12 bytes
Range: 000000000000 … FFFFFFFFFFFE
000000000000 is not a valid address but resets the original Node ID derived and hashed from the µC's unique MAC address if supported by µC otherwise: 000000000001

Example: `SetNodeIDAdd 0000BF260468`

Return value: **<ID>**
Format: 12 bytes
Range: 000000000000 … FFFFFFFFFFFE
Description: configured 6 byte Node ID of *swarm* node
if set ID = 000000000000 then default ID is returned

ReadNodeIDAdd <void>:

Description: Readback of configured Node ID of node connected to host
Parameters: void

Example: `ReadNodeIDAdd`
Return value: **<ID>**
Format: 12 bytes
Range: 000000000001 … FFFFFFFFFFFE
Description: configured 6 byte Node ID of *swarm* node

SetNodeType <TYPE>:

Description: Sets the type of node which will also be broadcasted in ID blinks
Parameters:
TYPE = 0 Passive node, responds to ranging requests if enabled, no data communication, no ranging reporting
TYPE = 1 Full active node, responds to ranging requests if enabled, initiates ranging requests, initiates data communication, reports ranging results, reports ID tag lists
TYPE = 2 Sniffer mode, only listens to packets and reports packets, does not initiate ranging nor responds to ranging requests

Format: 1 byte
Range: 0 … 2 corresponding to ASCII values ("0" … "2")

Example: `SetNodeType 2`

Return value: **<TYPE>**
Format: 1 byte
Range: 0…2 corresponding to ASCII values ("0" … "2")
Description: returning parameter which has been set
TYPE Type of node which has been set

SaveSettings:

Description: Saves all setting including Node ID permanently to EEPROM
Parameters: none

Example: `SaveSettings`

Return value: **<errorcode>**
Format: 1 byte
Range: 0…1 corresponding to ASCII values (“0” … “1”)
Description: Result of saving operation
errorcode = 0 Saving of all parameters successfully verified
errorcode = 1 Saving of parameters not successful; verification failed

RestoreSettings:

Description: Restores all parameter settings from EEPROM
Parameters: none

Example: `RestoreSettings`

Return value: **<errorcode>**
Format: 1 byte
Range: 0…1 corresponding to ASCII values (“0” … “1”)
Description: Result of restoring operation
errorcode = 0 Restoring of all parameters successful
errorcode = 1 Restoring parameters from EEPROM failed

ReadSettings:

Description: Reads current device configuration. First line is the number of following lines. All others state the name of parameter separated with ‘:’ and value. The value depends on parameter.

```
#<NumLines>
<ParameterName>:<Value>
<ParameterName>:<Value>
        …
```

Parameters: none

Example: `ReadSettings`

Return values: **<NumLines>**
Format: 4 bytes, first byte fixed „#“
Range: 000 … 255 corresponding to three ASCII values 30 … 39 (hex)
Description: Number of lines after this line

<ParameterName>
Format: ASCII
Description: Name of following parameter value.

<Value>
Format: Depends on parameter.

Example:

```
#009
FW_VER:ver1.6.1
MAC:000000000001
RANGING_BRDC:1
ID_BRDC:1
BRDC_INTERVAL:030
NODE_TYPE:1
PRIVACY:0
DNO:0
CSMA:1
```

SetFactorySettings:

Description: Reset device configuration factory settings.
Default configuration is:

MAC:000000000001.
Ranging broadcast enabled.
ID broadcast enabled.
ID broadcast interval 30s.
Node type => active.

Responds to ranging requests is true.
Data notification is enabled.
CSMA is enabled with seed (1).

Parameters: none

Example: `SetFactorySettings`

Return value: **<errorcode>**

Format:	1 byte
Range:	0…1 corresponding to ASCII values (“0” … “1”)
Description:	Result of writing operation to EEPROM
errorcode = 0	EEPROM successfully written
errorcode = 1	Writing parameters to EEPROM failed

4.4.2. Ranging Commands

SetPrivacyMode <ENABLE>:

Description: Enables and disables response to a received ranging request
Parameters: ENABLE = 0 Node will respond to ranging requests
ENABLE = 1 Node will not respond to ranging requests
Format: 1 byte
Range: 0 … 1 corresponding to ASCII values (“0” … “1”)

Example: `SetPrivacyMode 1`

Return value: **<ENABLE>**
Format: 1 byte
Range: 0…1 corresponding to ASCII values (“0” … “1”)
Description: returning parameter which has been set
ENABLE = 0 Privacy mode disabled, ranging response enabled
ENABLE = 1 Privacy mode enabled, no ranging response

RangeTo <ID>:

Description: Initiates an elementary ranging cycle to node with node ID <ID>
Parameters: <ID> 6 byte Node ID of ranging partner node
Format: 12 bytes

Example: `RangeTo 0000BF260468`

Return values: errorcode, ranging result, antenna

<Errorcode>
Format: 1 byte
Range: 0 … 4 corresponding to ASCII values (“0” … “4”)

Description: indicating status of ranging operation
Errorcode = 0: success → ranging result valid
Errorcode = 1: ranging to own ID
Errorcode = 2: ID out of range, no ACK
Errorcode = 3: ranging unsuccessful, ACK OK, then timeout
Errorcode = 4: only one ranging operation successful in diversity mode

<Ranging result>
Format: 7 bytes
Range: 0000.00 … 9999.99 ranging distance in meters
Description: returning the measured ranging distance in meters
Diversity off: ranging result with antenna <active>
Diversity on: minimum of 2 ranging results with antenna 0 and 1
If only one valid ranging result exists in diversity mode, this result is returned and Errorcode=4 is set

<Antenna>
Format: 1 byte
Range: 0 … 2 corresponding to ASCII values (“0” … “2”)

Description: returning the antenna used by non-initiating ranging partner
antenna = 0: antenna 0 has been used for ranging
antenna = 1: antenna 1 has been used for ranging
antenna = 2: both antennas have been used for ranging (diversity on only)

Remark:
The non-initiating ranging partner will respond by initially using antenna 0 and after completing the ranging request then switch to antenna 1 and vice versa. The used antenna will be transmitted back to the initiator as payload.
For swarm radios that do not support diversity the return value will be antenna=0

GetRangingResults <NodeID> <AGE>:

Description: Reports the received indirect ranging results between the selected node <NodeID> and other nodes with IDs <TargetID 1>, <TargetID 2>, … <TargetID n> which have a maximum age of AGE in seconds

Parameters: <NodeID>
Format: 12 bytes
Range: 000000000001 … FFFFFFFFFFFE

<AGE>
AGE = 000 Return all entries in Ranging Results List independent of age
AGE = 001-255 Maximum age for NodeIDList entries in seconds
Format: 3 bytes
Range: 000 … 255 corresponding to three ASCII values 30 … 39 (hex)

Example:
```
GetRangingResults 1F3C26041968 240
```

Return values:

<NumLines>
Format: 4 bytes, first byte fixed "#"
Range: 000 … 255 corresponding to three ASCII values 30…39 (hex)
Description: Number of Lines after this line

<NodeID>
Format: 12 bytes
Range: 000000000001 … FFFFFFFFFFFE
Description: 6 byte Node ID of selected *swarm* node

<AGE>
Format: 3 bytes
Range: 000 … 255 corresponding to three ASCII values 30…39 (hex)
Description: returning parameter which has been set
AGE Selected maximum age for ranging results

<RangingSets>
Format: 3 bytes
Range: 000 … 255 corresponding to three ASCII values 30…39 (hex)
Description: Number of ranging sets for NodeID in current ranging list

<TargetIDListEntry>
Format: 23 bytes
Range: AGE 000 … 255
TargetID 000000000001…FFFFFFFFFFFE
Distance [m] 0000.00…9999.99
Description: AGE Time since last seen, sorted last seen one first corresponding to three ASCII values 30…39 (hex) ":" as separator
Target ID Target ID node ID corresponding to twelve ASCII values ("0"…"1", "A" … "F")
Distance Distance between NodeID and TargetID in meters

Example:
```
#011
1F3C26041968
240
008
005:1F3CDD322123:0010.23
008:1F3CFF322133:0020.32
009:1F58DD322154:0026.34
009:1F3C31051999:0145.21
026:1F31051999C3:0003.22
098:1F318052001A:0299.34
129:1F3CDD3221EE:1201.12
239:1F3CDD3221FA:0050.26
```

BroadcastRangingResults <ENABLE>:

Description: Enables/Disables the broadcast transmission of ranging results after each successful ranging.

Parameters: ENABLE = 0 broadcast disabled
ENABLE = 1 broadcast enabled
Format: 1 byte
Range: 0 …1 corresponding to ASCII values (“0” … “1”)

Example: `BroadcastRangingResults 1`

Return value: **<ENABLE>**
Format: 1 byte
Range: 0 … 1 corresponding to ASCII values (“0” … “1”)
Description: returning parameter which has been set
ENABLE = 0 broadcast disabled
ENABLE = 1 broadcast enabled

DeleteAllRangingResults:

Description: Deletes all entries in Ranging Results List
Parameters: none

Example: `DeleteAllRangingResults`

Return values: **<NumberofDeletedEntries>**
Format: 3 bytes
Range: 000 … 255
Description: Number of deleted entries from ranging results list

Example: `240`

4.4.3. Data Communication Commands

EnableDataNotification <NOTIFY>:

Description: Enables and disables data notification
Parameters: NOTIFY = 0 Node will not trigger host when data paket has been received
NOTIFY = 1 Node will trigger host when data paket has been received
Format: 1 byte
Range: 0 … 1 corresponding to ASCII values (“0” … “1”)

Example: `EnableDataNotification 1`

Return value: **<NOTIFY>**
Format: 1 byte
Range: 0 …1 corresponding to ASCII values (“0” … “1”)
Description: returning parameter which has been set
NOTIFY Data notification enabled/disabled

SendDataTo <ID> <len> <data>:

Description: Sends <data> of length <len> to node <ID>
Parameters: <ID> 6 byte Node ID of ranging partner node
Format: 12 bytes
Range: 000000000001 … FFFFFFFFFFFE
corresponding to twelve ASCII values 30 … 39 (hex)

<len> length of payload in bytes (HEX)
Format: 2 bytes
Range: 01 … 80 (hex)
corresponding to two ASCII values (“0”…”9”, ”A” … “F”)

<data> payload to be transmitted
Format: 2 bytes <len> times 2 bytes of payload
Range: 00 … FF
corresponding to two ASCII values (“0”…”9”, ”A” … “F”)

Example: `SendDataTo 1F318052001A 02 FA13`

Return value: **<errorcode>**

Format: 1 byte
Range: 0 … 1 corresponding to ASCII values (“0” … “1”)

Description: indicating status of ranging operation
Errorcode = 0: success → data communication valid
Errorcode = 1: error: timeout; message could not be delivered

GetData <void>:

Description: Reads out transmitted data
Parameters: void

Example: `GetData`

Return values: Number of bytes, ID, Payload

<Number of bytes>
Format: 2 bytes
Range: 00…80 (hex)
corresponding to two ASCII values (“0”…”9”, ”A” … “F”)
Description: returns the number of bytes in pending message
Number of bytes = 00: no pending message available
Number of bytes = 01…80 (hex) number of bytes in message

<ID>
Format: 12 bytes
Range: 000000000001 … FFFFFFFFFFFE

corresponding to two ASCII values (“0”…”9”, ”A” … “F”)
Description: returns ID of node which sent message

<Payload>
Format: 2 bytes <Number of bytes> times 2 bytes of payload
Range: 00 … FF
corresponding to two ASCII values (“0”…”9”, ”A” … “F”)
Description: payload received

BroadcastData <len> <data>:

Description: Broadcasts <data> of length <len> to all nodes
Parameters: <len> length of payload in bytes (HEX)
Format: 2 bytes
Range: 01 … 80 (hex)
corresponding to two ASCII values (“0”…”9”, ”A” … “F”)

: <data> payload to be transmitted
Format: 2 bytes <len> times 2 bytes of payload
Range: 00 … FF
corresponding to two ASCII values (“0”…”9”, ”A” … “F”)

Example: `BroadcastData 02 FA13`

Return value: **<len>**
Format: 2 bytes
Range: 01 … 80 (hex)
corresponding to two ASCII values (“0”…”9”, ”A” … “F”)
Description: length of payload in bytes (HEX)

EnableRangingData <ENABLE>:

Description: Enables and disables the transmission of data from the ranging data buffer along with a ranging operation initiated by RangeTo.
Parameters: ENABLE = 0 Ranging data will not be transmitted with RangeTo command
ENABLE = 1 Ranging data will be transmitted with RangeTo command
Format: 1 byte
Range: 0 … 1 corresponding to ASCII values (“0” … “1”)

Example: `EnableRangingData 1`

Return value: **<ENABLE>**
Format: 1 byte
Range: 0…1 corresponding to ASCII values (“0”…”1”) (hex)
Description: returning parameter which has been set
ENABLE = 0 Ranging response disabled
ENABLE = 1 Ranging response enabled

FillRangingData <len><data>:

Description: Fills the ranging data buffer with <data> of length <len>. This data will be transmitted with the next RangeTo operation if EnableRangingData is <On>. The ranging data <data> is contained within the ranging packet itself.
Parameters: <len> length of ranging data payload in bytes (HEX)
Format: 2 bytes
Range: 01 … 74 (hex)
corresponding to two ASCII values (“0”…”9”, ”A” … “F”)

: <data> payload to be transmitted
Format: 2 bytes <len> times 2 bytes of payload
Range: 00 … FF
corresponding to two ASCII values (“0”…”9”, ”A” … “F”)

Example: `FillRangingData 0A FA13192F680426AE2345`

Return value: **<errorcode>**

Format:	1 byte
Range:	0 … 1 corresponding to ASCII values (“0” … “1”)
Description:	Status on ranging data buffer fill operation
Errorcode = 0:	successful
Errorcode = 1:	not successful

4.4.4. *swarm* radio Node Identification

SetBroadcastNodeID <ENABLE>:

Description: Enables and disables broadcast of Node ID blink packets.
Parameters: ENABLE = 0 Broadcast of Node ID blink packets disabled
ENABLE = 1 Broadcast of Node ID blink packets enabled
Format: 1 byte
Range: 0 … 1 corresponding to ASCII values (“0” … “1”)

Example: `SetBroadcastNodeID 1`

Return value: **<ENABLE>**
Format: 1 byte
Range: 0 …1 corresponding to ASCII values (“0” … “1”)
Description: returning parameter which has been set
ENABLE = 0 Broadcast of Node ID blink packets disabled
ENABLE = 1 Broadcast of Node ID blink packets enabled

SetBroadcastInterval <TIME>:

Description: Sets the broadcast interval in which the Node ID will be sent
Parameters: TIME Blink interval in seconds
Format: 2 bytes
Range: 01 … 99 corresponding to two ASCII values 30 … 39 (hex)

Example: `SetBroadcastInterval 02`

Return value: **<TIME>**
Format: 2 bytes
Range: 00…99 corresponding to two ASCII values 30…31 (hex)
Description: returning parameter which has been set
TIME Time interval for Node ID blink broadcasts

GetNodeIDList <AGE>:

Description: Reports the currently valid NodeIDList with time stamps that have a maximum age AGE in seconds
Parameters: AGE = 000 Delete all entries in NodeIDList
AGE = 001-255 Maximum age for NodeIDList entries in seconds
Format: 3 bytes
Range: 000 … 255 corresponding to three ASCII values 30 … 39 (hex)

Example: `GetNodeIDList 240`

Return values: **<NumLines>**
Format: 4 bytes, first byte fixed “#”
Range: 000 … 255 corresponding to three ASCII values 30…39 (hex)
Description: Number of Lines after this line

<AGE>
Format: 3 bytes
Range: 000 … 255 corresponding to three ASCII values 30…39 (hex)
Description: returning parameter which has been set
AGE Validity time for NodeIDList
If set parameter is 000, AGE returns the last set interval

<NumNodeIDs>
Format: 3 bytes
Range: 000 … 255 corresponding to three ASCII values 30…39 (hex)
Description: Number of NodeIDs in current NodeIDList

<NodeIDListEntry>
Format: 16 bytes
Range: 000 … 255:000000000001…FFFFFFFFFFFE
Description: Time since last seen, sorted last seen one first
corresponding to three ASCII values 30…39 (hex)

":" as separator
Node ID
corresponding to twelve ASCII values ("0"…"1", "A" … "F")

Example:

```
#010
240
008
005:1F3CDD322123
008:1F3CFF322133
009:1F58DD322154
009:1F3C26041968
026:1F31051999C3
098:1F318052001A
129:1F3CDD3221EE
239:1F3CDD3221FA
```

4.4.5. Air Interface Commands

SetCSMA <M>:

Description: Switches CSMA mode on and off and determines back-off factor for CSMA
Parameters: M = 0 CSMA off → ALOHA
M = 1 … 255 CSMA on, M = Back-off factor
Format: 1 byte
Range: 0 … 255 corresponding to ASCII values (“0” … “9”)

Example: `SetCSMA 0`

Return value: **<M>**
Format: 1 byte
Range: 0…255 corresponding to ASCII values (“0” … “9”)
Description: returning parameter which has been set
M = 0 CSMA off → ALOHA
M = 1 … 255 CSMA on, M = Back-off factor

SetDiversity <ENABLE>:

Description: Switches diversity mode for this node on and off
Parameters: ENABLE = 0 diversity off
ENABLE = 1 diversity on
Format: 1 byte
Range: 0 … 1 corresponding to ASCII values (“0” … “1”)

Example: `SetDiversity 1`

Return value: **<ENABLE>**
Format: 1 byte
Range: 0…1 corresponding to ASCII values (“0” … “1”)
Description: returning parameter which has been set
ENABLE = 0 diversity off
ENABLE = 1 diversity on

SetAntenna <A>:

Description: Selects the active antenna to be used for ranging operation if diversity is off
Parameters: A = 0 antenna 0 selected
A = 1 antenna 1 selected
Format: 1 byte
Range: 0 … 1 corresponding to ASCII values (“0” … “1”)

Example: `SetAntenna 1`

Return value: **<A>**
Format: 1 byte
Range: 0…1 corresponding to ASCII values (“0” … “1”)
Description: returning parameter which has been set
A = 0 antenna 0
A = 1 antenna 1

4.5. Format for Data Notification Messages

This chapter describes the communication structure for Data Notifications when data notification has been enabled.

Notification format: Data Notification Flag (fixed), ID

Data Notification Flag:
Format: 3 bytes
Content: 'DNO' 44 4E 4F (HEX)

ID:
Format: 12 bytes
Range: 000000000001 … FFFFFFFFFFFE
corresponding to two ASCII values ("0"…"9", "A" … "F")
Description: returns ID of node which sent message

Example: DNO:1F3CFF322133

4.6. API Default Settings

When starting the Swarm Radio the following default settings, representing an active node, are valid:

Broadcast Ranging Results :	on
ID Broadcast:	on
ID Broadcast Interval :	30s
Node Type:	1 (Full Active Node)
Privacy Mode:	off, respond to ranging requests
Data Notification:	on
CSMA:	on
Enable Ranging Data:	off

4.7. Setting for different Node Behaviours

The following parameter values are set when a specific node behaviour is selected:

Parameter	Passive (TYPE=0)	Active (TYPE=1)	Sniffer (TYPE=2)
Broadcast Ranging Results	off	on	off
ID Broadcast	on	on	off
ID Broadcast Interval	30 s	30 s	n/a
Node Type:	0	1	2
Privacy Mode	off	off	on
Data Notification	off	on	off
CSMA	off	on	off
Enable Ranging Data	off	off	off

5. Revision History

Date	Authors	Version	Description
2012-05-30	F. Schlichting	1.0	Initial version
2012-06-01	F. Schlichting	1.1	Feedback from SW added, LED UI specified
2012-06-06	F. Schlichting	1.2	Customer feedback added, API commands added, commands format specified, communication protocol specified
2012-09-24	F. Schlichting	1.3	Rename to *swarm* Ranging Demonstrator API, Expansion for greater No.of *swarm* nodes, ID broadcast blink, packet communication, ID List read out, option to deactivate ranging response
2012-09-28	F. Schlichting	1.4	Streamlined for nanoPAN 5375 Boards, product name change, text changes, functional changes, diversity removed
2012-10-08	F. Schlichting	1.5	GetRangingResults added, redefine node types, delete SetTagIDListParam command, add age to GetTagIDList, GetRangingResults, some commands renamed
2012-11-16	F. Schlichting	1.5.1	DeleteAllRangingResults, EnableDataNotification, Format for data notification messages added, minor definition changes
2013-01-18	F. Schlichting	1.6	BroadcastData, FillRangingData, SaveSettings, RestoreSettings, EnableRangingData added, ClearSendBuffer removed, default settings added, parameter values adjusted, rename document title, minor typing errors corrected, LED settings changed, “+++” command removed, return value for BroadcastData added, switch logic changed
2013-02-19	F. Schlichting	1.6.1	Text changes, Multi-line output format changed, API command set overview and categories added, commands ReadSettings and SetFactorySettings added, hardware specific content transferred to separate document, parameters changed in: SetCSMA, FillRangingData, SendDataTo, GetData, BroadcastData

End of Document

Life Support Policy

These products are not designed for use in life support appliances, devices, or systems where malfunction of these products can reasonably be expected to result in personal injury. Nanotron Technologies GmbH customers using or selling these products for use in such applications do so at their own risk and agree to fully indemnify Nanotron Technologies GmbH for any damages resulting from such improper use or sale.

Electromagnetic Interference / Compatibility
Nearly every electronic device is susceptible to electromagnetic interference (EMI) if inadequately shielded, designed, or otherwise configured for electromagnetic compatibility. To avoid electromagnetic interference and/or compatibility conflicts, do not use this device in any facility where posted notices instruct you to do so. In aircraft, use of any radio frequency devices must be in accordance with applicable regulations. Hospitals or health care facilities may be using equipment that is sensitive to external RF energy.
With medical devices, maintain a minimum separation of 15 cm (6 inches) between pacemakers and wireless devices and some wireless radios may interfere with some hearing aids. If other personal medical devices are being used in the vicinity of wireless devices, ensure that the device has been adequately shielded from RF energy. In a domestic environment this product may cause radio interference in which case the user may be required to take adequate measures.

CAUTION - Electrostatic Sensitive Device! Precaution should be used when handling the device in order to prevent permanent damage.

FCC User Information

Statement according to FCC part 15.19:
This device complies with Part 15 of the FCC Rules. Operation is subject to the following two conditions: (1) this device may not cause harmful interference, and (2) this device must accept any interference received, including interference that may cause undesired operation.

Statement according to FCC part 15.21:
Modifications not expressly approved by this company could void the user's authority to operate the equipment.

RF exposure:
The internal / external antennas used for this mobile transmitter must provide a separation distance of at least 20 cm from all persons and must not be co-located or operating in conjunction with any other antenna or transmitter.

Statement according to FCC part 15.105:
This equipment has been tested and found to comply with the limits for a Class A and Class B digital device, pursuant to Part 15 of the FCC Rules. These limits are designed to provide reasonable protection against harmful interference in a residential installation and against harmful interference when the equipment is operated in a commercial environment.

This equipment generates, uses, and can radiate radio frequency energy and, if not installed and used in accordance with the instructions as provided in the user manual, may cause harmful interference to radio communications. However, there is no guarantee that interference will not occur in a particular installation. Operation of this equipment in a residential area is likely to cause harmful interference in which case the user will be required to correct the interference at his or her own expense.

If this equipment does cause harmful interference to radio or television reception, which can be determined by turning the equipment off and on, the user is encouraged to try to correct the interference by one or more of the following measures: (1) reorient or relocate the receiving antenna, (2) increase the separation between the equipment and receiver, (3) connect the equipment into an outlet on a circuit different from that to the connected equipment, and (4) consult the dealer or an experienced technician for help.

About Nanotron Technologies GmbH

Nanotron provides reliable loss protection technology and solutions that are used to protect people and animals. Energy efficient, battery-powered wireless nodes are the key building blocks. These small devices create a Virtual Safety Zone which protects tagged people and animals. Robust wireless Chirp technology underpins nanotron's offering of chips, modules and loss protection software for indoor and outdoor environments world wide.

Headquartered in Berlin, Germany, *Nanotron Technologies GmbH* was founded in 1991.

Further Information

For more information about products from *Nanotron Technologies GmbH*, contact a sales representative at the following address:

Nanotron Technologies GmbH
Alt-Moabit 60
10555 Berlin, Germany
Phone: +49 30 399 954 – 0
Fax: +49 30 399 954 – 188
Email: sales@nanotron.com
Internet: www.nanotron.com